NAME

attr_remove, attr_removef − remove a user attribute of a filesystem object

C SYNOPSIS

#include <attr/attributes.h>

int attr_remove (const char *_path_, const char *_attrname_, int _flags_);

int attr_removef (int _fd_, const char *_attrname_, int _flags_);

DESCRIPTION

The **attr_remove** and **attr_removef** functions provide a way to remove previously created attributes from filesystem objects.

Path points to a path name for a filesystem object, and *fd* refers to the file descriptor associated with a file. If the attribute *attrname* exists, the attribute name and value will be removed from the filesystem object. The *flags* argument can contain the following symbols bitwise OR'ed together:

ATTR_ROOT
: Look for *attrname* in the **root** address space, not in the **user** address space. (limited to use by super-user only)

ATTR_DONTFOLLOW
: Do not follow symbolic links when resolving a *path* on an **attr_remove** function call. The default is to follow symbolic links.

attr_remove will fail if one or more of the following are true:

[ENOATTR] The attribute name given is not associated with the indicated filesystem object.

[ENOENT] The named file does not exist.

[EPERM] The effective user ID does not match the owner of the file and the effective user ID is not super-user.

[ENOTDIR] A component of the path prefix is not a directory.

[EACCES] Search permission is denied on a component of the path prefix.

[EINVAL] A bit was set in the *flag* argument that is not defined for this system call.

[EFAULT] *Path* points outside the allocated address space of the process.

[ELOOP] A path name lookup involved too many symbolic links.

[ENAMETOOLONG] The length of *path* exceeds {*MAXPATHLEN*}, or a pathname component is longer than {*MAXNAMELEN*}.

attr_removef will fail if:

[ENOATTR] The attribute name given is not associated with the indicated filesystem object.

[EINVAL] A bit was set in the *flag* argument that is not defined for this system call, or *fd* refers to a socket, not a file.

[EFAULT] *Attrname* points outside the allocated address space of the process.

[EBADF] *Fd* does not refer to a valid descriptor.

DIAGNOSTICS

On success, zero is returned. On error, −1 is returned, and *errno* is set appropriately.

SEE ALSO

attr(1), **attr_get**(3), **attr_list**(3), **attr_multi**(3), **attr_set**(3)